A Quantum Time-Space Lower Bound for the Counting Hierarchy[*]

Dieter van Melkebeek Thomas Watson
University of Wisconsin - Madison

February 2, 2008

Abstract

We obtain the first nontrivial time-space lower bound for quantum algorithms solving problems related to satisfiability. Our bound applies to MajSAT and MajMajSAT, which are complete problems for the first and second levels of the counting hierarchy, respectively. We prove that for every real d and every positive real ϵ there exists a real $c > 1$ such that either:

- MajMajSAT does not have a quantum algorithm with bounded two-sided error that runs in time n^c, or
- MajSAT does not have a quantum algorithm with bounded two-sided error that runs in time n^d and space $n^{1-\epsilon}$.

In particular, MajMajSAT cannot be solved by a quantum algorithm with bounded two-sided error running in time $n^{1+o(1)}$ and space $n^{1-\epsilon}$ for any $\epsilon > 0$.

The key technical novelty is a time- and space-efficient simulation of quantum computations with intermediate measurements by probabilistic machines with unbounded error. We also develop a model that is particularly suitable for the study of general quantum computations with simultaneous time and space bounds. However, our arguments hold for any reasonable uniform model of quantum computation.

1 Introduction

Satisfiability, the problem of deciding whether a given Boolean formula has at least one satisfying assignment, has tremendous practical and theoretical importance. It emerged as a central problem in complexity theory with the advent of NP-completeness in the 1970's. Proving lower bounds on the complexity of satisfiability remains a major open problem. Complexity theorists conjecture that satisfiability requires exponential time and linear space to solve in the worst case. Despite decades of effort, the best single-resource lower bounds for satisfiability on general-purpose models of computation are still the trivial ones – linear for time and logarithmic for space. However, since the late 1990's we have seen a number of results that rule out certain nontrivial combinations of time and space complexity.

One line of research [6, 7, 19, 5, 20], initiated by Fortnow, focuses on proving stronger and stronger time lower bounds for deterministic algorithms that solve satisfiability in small space. For subpolynomial (i.e., $n^{o(1)}$) space bounds, the current record states that no such algorithm can run in time n^c for any $c < 2\cos(\pi/7) \approx 1.8019$. A second research direction aims to strengthen the lower

[*]Research partially supported by NSF awards CCR-0133693 and CCF-0523680.

bounds by considering more powerful models of computation than the standard deterministic one. Diehl and Van Melkebeek [5] initiated the study of lower bounds for problems related to satisfiability on randomized models with bounded error. They showed that for every integer $\ell \geq 2$, Σ_ℓSAT cannot be solved in time n^c by subpolynomial-space randomized algorithms with bounded two-sided error for any $c < \ell$, where Σ_ℓSAT denotes the problem of deciding the validity of a given fully quantified Boolean formula with ℓ alternating blocks of quantifiers beginning with an existential quantifier. Σ_ℓSAT represents the analogue of satisfiability for the ℓth level of the polynomial-time hierarchy; Σ_1SAT corresponds to satisfiability. Proving nontrivial time-space lower bounds for satisfiability on randomized algorithms with bounded two-sided error remains open. Allender et al. [2] considered the even more powerful (but physically unrealistic) model of probabilistic algorithms with unbounded error[1]. They settled for problems that are even harder than Σ_ℓSAT for any fixed ℓ, namely MajSAT and MajMajSAT, the equivalents of satisfiability and Σ_2SAT in the counting hierarchy. MajSAT is the problem of deciding whether a given Boolean formula is satisfied for at least half of the assignments to its variables. MajMajSAT is the problem of deciding whether a given Boolean formula φ on disjoint variable sets x and y has the property that for at least half of the assignments to x, φ is satisfied for at least half of the assignments to y. Recall that Toda [16] proved that the polynomial-time hierarchy reduces to the class PP, which represents polynomial-time probabilistic computations with unbounded two-sided error and forms the first level of the counting hierarchy. Apart from dealing with harder problems, the quantitative strength of the lower bounds by Allender et al. is also somewhat weaker. In particular, they showed that no probabilistic algorithm can solve MajMajSAT in time $n^{1+o(1)}$ and space $n^{1-\epsilon}$ for any positive constant ϵ. We refer to [13] for a detailed survey of the past work on time-space lower bounds for satisfiability and related problems, including a presentation of the Allender et al. lower bound that is slightly different from the original one.

In this paper we study the most powerful model that is considered physically realistic, namely quantum algorithms with bounded error. We obtain the first nontrivial time-space lower bound for quantum algorithms solving problems related to satisfiability. In the bounded two-sided error randomized setting, the reason we can get lower bounds for Σ_ℓSAT for $\ell \geq 2$ but not for $\ell = 1$ relates to the fact that we know efficient simulations of such randomized computations in the second level of the polynomial-time hierarchy but not in the first level. In the quantum setting the situation is worse: we know of no efficient simulations in any level of the polynomial-time hierarchy. The best simulations to date are due to Adleman et al. [1], who showed that polynomial-time quantum computations with bounded two-sided error can be simulated in PP. Building on this connection, we bring the lower bounds of Allender et al. to bear on bounded-error quantum algorithms. Our main result shows that either a time lower bound holds for quantum algorithms solving MajMajSAT or a time-space lower bound holds for MajSAT.

Theorem 1. *For every real d and every positive real ϵ there exists a real $c > 1$ such that either:*

- MajMajSAT *does not have a quantum algorithm with bounded two-sided error that runs in time n^c, or*
- MajSAT *does not have a quantum algorithm with bounded two-sided error that runs in time n^d and space $n^{1-\epsilon}$.*

[1]Throughout this paper, we use the word "randomized" for the bounded-error setting and "probabilistic" for the unbounded-error setting.

As a corollary, we obtain a single time-space lower bound for MajMajSAT.

Corollary 1. MajMajSAT *cannot be solved by a quantum algorithm with bounded two-sided error running in time* $n^{1+o(1)}$ *and space* $n^{1-\epsilon}$ *for any* $\epsilon > 0$.

Unlike in the deterministic and randomized cases, it is not obvious how to define a model of quantum computation that allows us to accurately measure both time and space complexity. The existing models give rise to various issues. For example, intermediate measurements play a critical role as they are needed for time-space efficient simulations of randomized computations by quantum computations. Several of the known models only allow measurements at the end of the computation but not during the computation. As another example, the classical time-space lower bounds hold for models with random access to the input and memory. This makes the lower bounds more meaningful as they do not exploit artifacts due to sequential access. Extending the standard quantum Turing machine model [3] to accommodate random access leads to complications that make the model inconvenient to work with. In Section 2 we discuss these and other issues in detail, and we survey the known models from the literature. We present a model that addresses all issues and is capable of efficiently simulating all other uniform models that may be physically realizable in the foreseeable future. Thus, lower bounds in our model reflect true problem hardness.

The main technical novelty for establishing Theorem 1 consists of an improved time- and space-efficient simulation of quantum computations by unbounded-error probabilistic computations. The previously known simulations, such as the one by Adleman et al., do not deal with intermediate measurements in a space-efficient way. We show how to cope with intermediate measurements in a space-efficient way and without loss in running time. Our construction works even when the sequence of local quantum operations can depend on previous measurement outcomes; i.e., we handle more powerful models than uniform quantum circuits. Our simulation makes use of a result on approximating quantum gates due to Solovay and Kitaev [11]. Theorem 1 follows from our simulation and the Allender et al. lower bound. The quantitative strength of our lower bound derives from the latter; our translation does not induce any further weakening.

The rest of this paper is organized as follows. We start with a discussion of the model in Section 2, we derive our results in Section 3, and we conclude with some open problems in Section 4. Throughout we assume basic background in quantum computation; see for example [14, 12].

2 Models of Quantum Computation

In this section we develop the model that we use for the exposition of our arguments. Section 2.1 contains a discussion of the issues that arise in choosing a model of quantum computation that accurately reflects time and space complexity. In Section 2.2 we describe how previously studied models fit into our taxonomy. We motivate and precisely define our chosen model in Section 2.3. Although we consider the development of such a model as a contribution of our paper, the crux of our main result can be understood at an abstract level. As such, a reader who would like to quickly get to the heart of our paper can skip Section 2.

2.1 Issues

Our model should capture the notion of a quantum algorithm as viewed by the computer science and physics communities and allow us to accurately measure the resources of time and space. For example, the model should allow us to express important quantum algorithms such as Shor's [15] and Grover's [9] in a way that is natural and faithfully represents their complexities. This forms the overarching issue in choosing a model. Below we discuss eight specific aspects of quantum computation models and describe how the corresponding issues are handled in the classical setting.

Sublinear space bounds. Many algorithms have the property that the amount of work space needed is less than the size of the input. Models such as one-tape Turing machines do not allow us to accurately measure the space usage of such algorithms because they charge for the space required to store the input. In the deterministic and randomized settings, sublinear space bounds are accommodated by considering Turing machines with a read-only input tape that does not count toward the space bound and read-write work tapes that do. In the quantum setting, we need a model with an analogous capability.

Random access to the input and memory. In order to accurately reflect the complexity of computational problems, our model should include a mechanism for random access, i.e., the ability to access any part of the input or memory in a negligible amount of time (say, linear in the length of the address). For example, there is a trivial algorithm for the language of palindromes that runs in quasilinear time and logarithmic space on standard models with random access, but the time-space product of any traditional sequential-access Turing machine deciding palindromes is at least quadratic. The latter result does not reflect the complexity of deciding palindromes, but rather exploits the fact that sequential-access machines may have to waste a lot of time moving their tape heads back and forth. Classical Turing machines can be augmented with a mechanism to support random access; our quantum model should also have such a mechanism.

Intermediate measurements. Unlike the previous two issues, intermediate measurements are specific to the quantum setting. In time-bounded quantum computations, it is customary to assume that all measurements occur at the end. This is because intermediate measurements can be postponed by introducing ancilla qubits to store what would be the result of the measurement, thus preventing computation paths with different measurement outcomes from interfering with each other. However, this has a high cost in space – a computation running in time t may make up to t measurements, so the space overhead could be as large as t, which could be exponential in the original space bound. Hence, to handle small space bounds our model should allow intermediate measurements. Indeed, this is crucial for our model to meet the expectation of being at least as strong as randomized algorithms with comparable efficiency parameters; the standard way to "flip a coin" in the quantum setting is to apply a Hadamard gate to a qubit in a basis state and then measure it. Also, many quantum algorithms, such as Shor's factoring algorithm, are naturally described using intermediate measurements.

We also need to decide which measurements to allow. Projective measurements in the computational basis are the most natural choice. Should we allow projective measurements in other bases? How about fully general measurements (see Section 2.2.3 in [14]), where the measurement operators need not be projections? General measurements can be performed by introducing ancilla qubits (at a cost in space), performing a change of basis (at a cost in time), and doing a projective measurement in the computational basis, one qubit at a time. It is reasonable to charge the complexity of these operations to the algorithm designer, so we are satisfied with allowing only single-qubit measurements in the computational basis.

Obliviousness to the computation history. Computations proceed by applying a sequence of local operations to data. We call a computation *nonoblivious* if at each step, which local operation to use and which operands to apply it to may depend on the computation history. A generic deterministic Turing machine computation is nonoblivious. We can view each state as defining an operation on a fixed number of tape cells, where the operands are given by the tape head locations. In each step, the outcome of the applied operation affects the next state and tape head locations, so both the operation and the operands can depend on the computation history. In contrast, a classical circuit computation is oblivious because neither the operation (gate) nor the operands (wires connected to the gate inputs) depend on the computation history (values carried on the wires).

In the randomized and quantum settings, the notion of a computation history becomes more complicated because there can be many computation paths. In the randomized setting, applying a randomized operation to a configuration may split it into a distribution over configurations, and the randomized Turing machine model allows the next state and tape head locations to depend on which computation path was taken. In the quantum setting, applying a quantum operation to a basis state may split it into a superposition over several basis states, and general nonoblivious behavior would allow the next operation and operands to depend on which computation path was taken. However, it is unclear whether such behavior is physically realizable, as currently envisioned technologies all select quantum operations classically. An intermediate notion of nonobliviousness, where the operations and operands may depend on previous measurement outcomes but not on the quantum computation path, does seem physically realistic.

Classical control. There is a wide spectrum of degrees of interaction between a quantum computation and its classical control. On the one hand, one can imagine a quantum computation that is entirely "self-sufficient," other than the interaction needed to provide the input and observe the output. On the other hand, one can imagine a quantum computation that is guided classically every step of the way. Self-sufficiency is inherent to computations that are nonoblivious to the quantum computation path, whereas measurements are inherently classically controlled operations. Incorporating intermediate measurements into computations that are nonoblivious to the quantum computation path would require some sort of global coordination among the quantum computation paths to determine when a measurement should take place.

Syntax. Our model should be syntactic, meaning that identifying valid programs in the model is decidable. If we are interested in bounded-error computations, then we cannot hope to decidably distinguish programs satisfying the bounded-error promise from those that do not. However, we should be able to distinguish programs that evolve according to the postulates of quantum mechanics from those that do not. Allowing nonobliviousness to the quantum computation path complicates this syntax check. If different components of the superposition can undergo different unitary operations then the overall operation is not automatically unitary, due to interference. Extra conditions on the transition function are needed to guarantee unitarity.

Complexity of the transition amplitudes. Care should be taken in specifying the allowable transition amplitudes. In the randomized setting, it is possible to solve undecidable languages by encoding the characteristic sequences of these languages in the transition probabilities. This problem is usually handled by using a certain universal set of elementary randomized operations, e.g., an unbiased coin flip. In the quantum setting, the same problem arises with unrestricted amplitudes. Again, one can solve the problem by restricting the elementary quantum operations to a universal set. However, unlike in the randomized setting, there is no single standard universal set like the unbiased coin flip with which all quantum algorithms are easy to describe. Algorithm designers should be allowed to use arbitrary local operations provided they do not smuggle hard-to-compute information into the amplitudes.

Absolute halting. In order to measure time complexity, we should use a model that naturally allows any algorithm to halt absolutely within some time bound t. In the randomized setting, one can design algorithms whose running times are random variables and may actually run forever. We can handle such algorithms by clocking them, so that they are forced to halt within some fixed number of time steps. Our quantum model should provide a similar mechanism.

2.2 Earlier Models

Now that we have spelled out the relevant issues and criteria, we consider several previously studied models as candidates.

Bernstein and Vazirani [3] laid the foundations for studying quantum complexity theory using quantum Turing machines. Their model uses a single tape and therefore cannot handle sublinear space bounds. Like classical one-tape Turing machines, their model is sequential-access. It does not allow intermediate measurements. On the other hand, their model is fully nonoblivious: the transition function produces a superposition over basis configurations, and the state and tape head location may be different for different components of the superposition. Their model represents the self-sufficient extreme of the classical control spectrum. In their paper, Bernstein and Vazirani prove that their model is syntactic by giving a few orthogonality constraints on the entries of the transition function table that are necessary and sufficient for the overall evolution to be unitary. These conditions are somewhat unnatural, and can be traced back to the possibility of nonobliviousness to the quantum computation path. Bernstein and Vazirani restrict the transition amplitudes by requiring that the first k bits of each amplitude are computable deterministically in time poly(k). Their model is nontrivial to clock; they require that the transition function be designed in such a way that the machine always halts, meaning that it reaches a superposition in which all non-halting basis configurations have zero amplitude. Bernstein and Vazirani detail how to design such mechanisms.

In [18], Watrous considers a model similar to Bernstein and Vazirani's, but with one read-write work tape and a read-only input tape not counting toward the space bound. The model naturally allows for sublinear space bounds, but it is still sequential-access. It allows intermediate measurements but only for the halting mechanism: a special register is measured after each time step, with the outcome indicating "halt and output 1", "halt and output 0", or "continue". The model is nonoblivious like the Bernstein-Vazirani model. It has more classical interaction due to the halting mechanism, but this is arguably not "classical control." The syntax conditions on the transition function are similar to those for the Bernstein-Vazirani model. The results in [18] require the transition amplitudes to be rational, which is somewhat unappealing since one may often wish to use Hadamard gates, which have irrational amplitudes. Similar to the Bernstein-Vazirani model, the model is nontrivial to clock. In fact, the results in [18] rely on counting an infinite computation as a rejection.

The main issue with the above models for our purposes is their sequential-access nature. It is possible to handle this problem by imposing a random-access mechanism. However, the conditions on the entries of the transition function table characterizing unitary evolution become more complicated and unnatural, making the model inconvenient to work with. Again, the culprit is the nonobliviousness to the quantum computation path. Since this behavior does not appear to be physically realizable in the foreseeable future anyway, the complications arising from it are in some sense unjustified.

In [17], Watrous considers a different model of space-bounded quantum computation. This model is essentially a classical Turing machine with an additional quantum work tape and a fixed-size quantum register. Sublinear space bounds are handled by charging for the space of the classical work tape and the quantum work tape but not the input tape. All three tape heads move sequentially. This model handles intermediate measurements. It is oblivious to the quantum computation path; the state and tape head locations cannot be in superposition with the contents of the quantum work tape. However, the computation is nonoblivious to the classical computation history, including the measurement outcomes. The finite control is classical; in each step it selects a quantum operation and applies it to the combination of the qubit under the quantum work tape head together with the fixed-size register. The register is needed because there is only one head on the quantum work tape, but a quantum operation needs to act on multiple qubits to create entanglement. The allowed operations come from the so-called quantum operations formalism (see Chapter 8 of [14]), which encompasses unitary operations and general measurements, as well as interaction with an external environment. Each quantum operation produces an output from a finite alphabet – the measurement outcome in the case of a measurement. This outcome influences the next (classical) transition. This model is syntactic just like classical Turing machines, with the additional step of testing that each quantum operation satisfies the definition of a valid quantum operation. For his constructions, Watrous needs the transition amplitudes to be algebraic. This model is trivial to clock, since all the control is done classically and thus the machine can halt in a fixed number of steps, just as in the classical setting.

The latter model is convenient to work with since the essence of the quantum aspects of a computation are isolated into local operations that are chosen classically and applied to a simple quantum register. This models the currently envisioned realizations of quantum computers. We adopt this model for the exposition of our results, but we need to make some modifications in order to address the following issues.

- Algorithms like Grover's require quantum access to the input, i.e., an operation that allows different basis states in a superposition to access different bits of the input simultaneously. On inputs of length n, this is done with a query gate that effects the transformation $|i\rangle|b\rangle \mapsto |i\rangle|b \oplus x_i\rangle$ where $i \in \{0,1\}^{\lceil \log n \rceil}$, $b \in \{0,1\}$, and x_i is the ith bit of the input. The model from [17] does not have such an operation and thus cannot express algorithms like Grover's. While this operation seems no more physically realistic than nonobliviousness to the quantum computation path if we view the input as stored in a classical memory, it does make sense when the input is actually the output of another computation. For these reasons, we include such an operation in our model.

- We want our model to have random access to emphasize the fact that our time-space lower bound does not exploit any model artifacts due to sequential access. We can make the model from [17] random-access by allowing each of the tape heads to jump in unit time to a location whose address we have classically computed, just as can be done for deterministic and randomized Turing machines.

- The quantum operations used in the model from [17] are more general than we wish to consider. Since we are focusing on the computational aspects of the model, we choose to restrict the set of allowed operations to unitary operations and projective measurements in the computational basis. The quantum operations formalism models the evolution of open quantum systems, which is of information-theoretic rather than algorithmic concern and can be simulated with unitary operations by introducing an additional "environment" system at a cost in space.
- The restriction to algebraic transition amplitudes is unnecessary in the present setting. We feel that a reasonable way to restrict the amplitudes is the one chosen by Bernstein and Vazirani; i.e., the first k bits of each amplitude should be computable deterministically in time poly(k).

2.3 Our Model

For concreteness, we now describe and motivate the particular model we use for the exposition of our arguments. Our model addresses all the issues listed in Section 2.1, and is an adaptation of Watrous's model from [17], as described at the end of Section 2.2.

In terms of obliviousness, our model corresponds to the physically realistic middle ground where a classical mechanism determines which quantum operation to apply based on the previous measurement outcomes but independent of the actual quantum computation path. Our arguments are robust with respect to the details of the model as long as it has the latter property. In particular, we can handle uniform quantum circuits. Our results also hold for more general models allowing nonobliviousness to the quantum computation path, but this requires more technical work; see the remarks in Section 3.4.

2.3.1 Model Definition

We define a quantum Turing machine as follows. There are three semi-infinite tapes: the input tape, the classical work tape, and the quantum work tape. Each cell on the input tape holds one bit or a blank symbol. Each cell on the classical work tape holds one bit. Each cell on the quantum work tape holds one qubit. The input tape contains the input, a string in $\{0,1\}^n$, followed by blanks, and the classical and quantum work tapes are initialized to all 0's. There are a fixed number of tape heads, each of which is restricted to one of the three tapes. There may be multiple heads moving independently on the same tape.

The finite control, the operations on the classical work tape, and all head movements are classical; each operation on the quantum work tape can be either a unitary operation or a single-qubit projective measurement in the computational basis. In each step of the computation, the finite control of the machine is in one of a finite number of states. Each state has an associated classical function, which is applied to the contents of the cells under the heads on the classical work tape, and an associated quantum operation, which is applied to the contents of the cells under the heads on the quantum work tape. The next state of the finite control and the head movements are determined by the current state, the contents of the cells under the input tape heads and classical work tape heads at the beginning of the computation step, and the measurement outcome if the quantum operation was a measurement.

Each head moves left one cell, moves right one cell, stays where it is, or jumps to a new location at a precomputed address that is written on the classical work tape between two of the classical work tape heads. The latter type of move is classical random access. We also allow "quantum random access" to the input by optionally performing a query that effects the transformation $|i\rangle|b\rangle \mapsto |i\rangle|b \oplus x_i\rangle$ on the qubits between two of the quantum work tape heads, where $i \in \{0,1\}^*$ is an address located on the quantum work tape, $b \in \{0,1\}$, and x_i is the ith bit of the input of length n or 0 if $i > n$.

Among the states of the finite control are an "accept" state and a "reject" state, which cause the machine to halt. Although not needed in this paper, the machine can be augmented with a one-way sequential-access write-only classical output tape in order to compute nonboolean functions.

Let us motivate our model definition. In terms of physical computing systems, the input tape corresponds to an external input source, the classical work tape corresponds to classical memory, and the quantum work tape corresponds to quantum memory. The bits and qubits under the heads correspond to the data being operated on in the CPU.

We use multiple heads on each tape for several reason. One reason is that creating entanglement requires multiple-qubit operations and hence multiple quantum work tape heads. Another reason is that having multiple heads offers a convenient way of formalizing random access. Since we are studying how algorithm performance scales with the input size, addresses have non-constant length and thus cannot fit under the tape heads all at once. A number of mechanisms are possible for indicating where an address is stored for random access. The one we have chosen, namely that the address is delimited by two tape heads, is artificial and is chosen only for convenience because it makes the model simple and clean. Another possible mechanism is to associate with each head a special *index tape* used for writing addresses; see [13] for a discussion of this type of model.

A minor issue arises with our multiple head approach: an operation on a work tape may not be well-defined if two of the heads are over the same cell. Rather than requiring programs to avoid this situation, which would make the model non-syntactic, we can just assume that no operation is performed on the violating work tape when this situation arises.

We allow the heads to move sequentially because if we only allowed random access, then constructing an address would require storing a pointer to the location where that address is stored. The pointer would have have nonconstant size, so we would need a pointer to that pointer, and so on. This chicken-and-egg problem does not appear in physical computing systems, and we explicitly avoid it by allowing sequential traversal of memory without having to "remember" where the head is.

2.3.2 Complexity Classes

The running time of a quantum Turing machine at input length n is the maximum over all inputs of length n and over all computation paths of the number of steps before the machine halts. The space usage is the maximum over all inputs of length n and over all computation paths of the largest address of a classical work tape head or quantum work tape head during the computation. Either the time or the space may be infinite. Note that we maximize over *all* computation paths, even ones that occur with probability 0 due to destructive interference.

Our definition of space usage allows the space to be exponential in the running time, since in time t a machine can write an address that is exponential in t and move a head to that location using the random-access mechanism. However, the space usage can be reduced to at most the running time with at most a polylogarithmic factor increase in the latter by compressing the data and using an appropriate data structure to store (old address, new address) pairs. (See Section 2.3.1 of [13] for a similar construction.)

We are now set up to define quantum complexity classes within our model.

Definition. $\mathrm{BQTISP}(t,s)$ *is the class of languages L such that for some quantum Turing machine M running in time $O(t)$ and space $O(s)$,*

- *if $x \in L$ then $\Pr(M \text{ accepts } x) \geq \frac{2}{3}$, and*
- *if $x \notin L$ then $\Pr(M \text{ accepts } x) \leq \frac{1}{3}$.*

We also require that for each entry in the matrix representation of each unitary operation of M in the computational basis, the first k bits of the real and imaginary parts are computable deterministically in time $\mathrm{poly}(k)$. We define $\mathrm{BQTIME}(t)$ similarly but without the space restriction.

As evidence in support of our model of choice, we note that the following results hold in our model.

- Recall that $\mathrm{BPTISP}(t,s)$ is the class of languages solvable in time $O(t)$ and space $O(s)$ by a randomized algorithm with error probability at most 1/3. Then $\mathrm{BPTISP}(t,s) \subseteq \mathrm{BQTISP}(t,s)$ holds because a quantum algorithm in our model can directly simulate a randomized algorithm; the only issue is producing unbiased coin flips. For this, the simulation can apply a Hadamard gate to one qubit on the quantum work tape and then measure it. This qubit can be reused to generate as many random bits as needed.
- Grover's algorithm shows that $\mathrm{OR} \in \mathrm{BQTISP}(n^{1/2} \cdot \mathrm{polylog}(n),\ \log n)$, where OR denotes the problem of computing the disjunction of the n input bits.
- Shor's algorithm shows that a nontrivial factor of an integer of bit length n can be computed in time $O(n^3 \cdot \mathrm{polylog}(n))$ and space $O(n)$ with error probability at most 1/3 in our model.

3 Time-Space Lower Bound

In this section we prove our results. Section 3.1 contains an outline of the two main steps of the proof. In Section 3.2 we argue that we can restrict our attention to a special case of our model using a finite universal set of gates. We show in Section 3.3 how to efficiently simulate this special case on unbounded-error probabilistic algorithms.

3.1 Results and Proof Outline

Using the notation introduced in Section 2.3.2, we can formalize Theorem 1 and Corollary 1 as follows.

Theorem 1 (restated). *For every real d and every positive real ϵ there exists a real $c > 1$ such that either:*

- $\mathrm{MajMajSAT} \notin \mathrm{BQTIME}(n^c)$*, or*
- $\mathrm{MajSAT} \notin \mathrm{BQTISP}(n^d, n^{1-\epsilon})$.

Corollary 1 (restated). *For all $\epsilon > 0$,* MajMajSAT $\notin$ BQTISP$(n^{1+o(1)}, n^{1-\epsilon})$.

Theorem 1 follows immediately from the following two results. The first gives a lower bound for MajSAT and MajMajSAT on unbounded-error probabilistic algorithms, and the second translates this lower bound to the quantum setting by giving a time- and space-efficient simulation of quantum algorithms by unbounded-error probabilistic algorithms. Recall that PTISP(t, s) denotes the class of languages decidable by unbounded-error probabilistic algorithms running in time $O(t)$ and space $O(s)$.

Lemma 1 (Allender et al. [2]). *For every real d and every positive real ϵ there exists a real $c > 1$ such that either:*

- MajMajSAT $\notin$ PTIME(n^c)*, and*
- MajSAT $\notin$ PTISP$(n^d, n^{1-\epsilon})$.

Lemma 2. *For all sufficiently constructible $t \geq \log n$ and $s \geq \log n$,*

$$\text{BQTISP}(t, s) \subseteq \text{PTISP}(t \cdot \text{polylog}(t),\ s + \text{polylog}(t)).$$

Lemma 2 is our main technical contribution, and its proof occupies the remainder of Section 3. The first step is to show that we can assume without loss of generality that our model only uses a certain finite universal set of quantum gates. A key ingredient is the Solovay-Kitaev theorem [11], which shows how to approximate any single-qubit unitary gate to within ϵ in the 2-norm sense using only polylog$(1/\epsilon)$ gates from a finite universal set. The efficiency afforded by the Solovay-Kitaev theorem is critical for obtaining our lower bound.

The second step is to simulate this special case of our model time- and space-efficiently with unbounded-error probabilistic algorithms. Our strategy builds on known simulations of quantum computations without intermediate measurements by probabilistic machines with unbounded error [1, 8]. The basic idea of these simulations is to write the final amplitude of a basis state as a simple linear combination of #P functions, where each #P function counts the number of quantum computation paths leading to that state with a certain path amplitude. Taking advantage of our choice of universal set, we can use simple algebraic manipulations to express the probability of acceptance as the difference between two #P functions, up to a simple common scaling factor. Standard techniques then result in a time- and space-efficient simulation by an unbounded-error probabilistic machine.

The above approach only handles unitary operations with one final measurement. To handle intermediate measurements, we first adapt this approach to capture the probability of observing any particular sequence of measurement outcomes. The acceptance probability can then be expressed as a sum over all sequences of measurement outcomes that lead to acceptance, where each term is the scaled difference of two #P functions. We can combine those terms into a single one using the closure of #P under uniform exponential sums. However, the usual way of doing this – nondeterministically guess and store a sequence and then run the computation corresponding to that sequence – is too space-inefficient. To address this problem, we note that the crux of the construction corresponds to multiplying two #P functions on the same input. The standard approach runs the two computations in sequence, accepting iff both accept. We argue that we can run these two computations *in parallel* and keep them in synch so that they access each bit of the guessed sequence at the same time, allowing us to reference each bit only once. We can then guess each bit when needed during the final simulation and overwrite it with the next guess bit, allowing us to meet the space constraint.

Regarding the conditions in Lemma 2, we assume t and s are at least logarithmic so that they dominate any logarithmic terms arising from indexed access to the input. We henceforth ignore the technical constructibility constraints on t and s. For Theorem 1, we only need to consider "ordinary" polynomially-bounded functions, which are computable in time polynomial in the length of the output written in binary, which is sufficient for our purposes.

3.2 Efficient Approximation With a Universal Set

A $\mathrm{BQTISP}(t, s)$ computation can be viewed as applying a sequence of $O(t)$ classically selected quantum gates to a register of $O(s)$ qubits. There are three types of gates:

- Unitary gates selected from a finite library of gates associated with the machine.
- Query gates, which effect the transformation $|i\rangle|b\rangle \mapsto |i\rangle|b \oplus x_i\rangle$, where i is an index into the input x.
- Measurement gates, which perform a single-qubit projective measurement in the computational basis.

The first step in the proof of Lemma 2 is to show that we can restrict our attention to machines whose library is a fixed universal set. It is well-known that every unitary transformation can be effected exactly using CNOT gates and single-qubit gates. Defining $\mathrm{BQTISP}'(t, s)$ to be $\mathrm{BQTISP}(t, s)$ with the restriction that each gate in the library either is CNOT or acts on only one qubit, we have the following.

Lemma 3. *For all t and all s,*

$$\mathrm{BQTISP}(t, s) = \mathrm{BQTISP}'(t, s).$$

For completeness, we sketch a proof of Lemma 3 in Appendix B.

It is also well-known that finite universal sets exist which can approximate any unitary operation to arbitrary accuracy. We say that a single-qubit unitary operation $\widetilde{U}$ *ϵ-approximates* a single-qubit unitary operation U if $||\widetilde{U} - e^{i\theta}U|| \leq \epsilon$ for some (irrelevant) global phase factor $e^{i\theta}$. We say that a set S of single-qubit unitary gates is *universal for single-qubit unitary gates* if for all single-qubit unitary gates U and all $\epsilon > 0$ there is a sequence $\widetilde{U}_1, \ldots, \widetilde{U}_\ell$ of gates from S such that the operation $\widetilde{U}_1 \cdots \widetilde{U}_\ell$ ϵ-approximates U.

We use the fact that the set $\{F, H\}$ is universal for single-qubit unitary gates[2], where H is the Hadamard gate and

$$F = \begin{bmatrix} 1 & 0 \\ 0 & \frac{3}{5} + \frac{4}{5}i \end{bmatrix}$$

in the computational basis. We can restrict our attention to quantum Turing machines with library $\{\mathrm{CNOT}, F, H\}$ by replacing each single-qubit gate in the library of a BQTISP' machine with an approximation using F and H gates.

[2]This is shown on page 196 in [14] with a relative phase shift by $\pi/4$ instead of the F gate; however, F is a relative phase shift by an irrational multiple of 2π and hence can approximate the $\pi/4$ phase shift to arbitrary accuracy.

We need to satisfy the following requirements:

- The transformation should not increase the number of gates applied by too much.
- The new sequence of gates should still be efficiently computable by a classical algorithm.
- The probability an input x is accepted should not change by too much when we apply the transformation.

The following key theorem allows us to meet these constraints.

Lemma 4 (Solovay and Kitaev [11]). *If S is universal for single-qubit unitary gates and is closed under adjoint, then for all single-qubit unitary gates U and all $\epsilon > 0$ there is a sequence of at most $\mathrm{polylog}(1/\epsilon)$ gates from S that ϵ-approximates U. Moreover, such a sequence can be computed deterministically in time $\mathrm{polylog}(1/\epsilon)$ provided the first k bits of the matrix entries of U and the gates in S are computable in time $\mathrm{poly}(k)$.*

The proof by Solovay and Kitaev gives an algorithm for computing an approximation in time $\mathrm{polylog}(1/\epsilon)$, ignoring the complexity of arithmetic (see [4] and Section 8.3 of [12]). We cannot do exact arithmetic since the entries of our gates may require infinitely many bits to specify, but in each step of the algorithm it suffices to work with $\mathrm{poly}(\epsilon)$-approximations to all of the matrices. Computing each matrix entry to $O(\log(1/\epsilon))$ bits suffices for this because the matrices have only constant size. By our complexity constraint on the transition amplitudes and the fact that the entries of F and H are also efficiently computable, this incurs only a $\mathrm{polylog}(1/\epsilon)$ time (and space) overhead.

We argue that approximating each single-qubit unitary gate to within $\Theta(1/t)$ ensures that the probability of acceptance of a quantum Turing machine running in time t only changes by a small amount. Note that our model is nonoblivious to measurement outcomes, so there may be exponentially many classical computation paths corresponding to the different measurement outcomes. A simple union bound over these paths does not work since it would require exponentially small precision in the approximations, which we cannot afford. However, the approximation errors are relative to the probability weights of the paths. As a result, the overall error cannot grow too large.

Define $\mathrm{BQTISP}''(t, s)$ to be $\mathrm{BQTISP}(t, s)$ with the restriction that the library of gates is $\{\mathrm{CNOT}, F, F^\dagger, H, I\}$. We include "identity gates" I for technical reasons – we need to allow computation steps that do not change the state of the quantum tape. We include $F^\dagger$ gates because the Solovay-Kitaev theorem requires the universal set to be closed under adjoint.

Lemma 5. *For all sufficiently constructible t and all s,*

$$\mathrm{BQTISP}'(t, s) \subseteq \mathrm{BQTISP}''(t \cdot \mathrm{polylog}(t),\ s + \mathrm{polylog}(t)).$$

We defer the proof of Lemma 5 to Appendix A.

3.3 Efficient Probabilistic Simulation

We now prove Lemma 2. By Lemmas 3 and 5, it suffices to show that

$$\mathrm{BQTISP}''(t, s) \subseteq \mathrm{PTISP}(t,\ s + \log t).$$

So consider a language $L \in \mathrm{BQTISP}''(t, s)$ and an associated quantum Turing machine M. We fix an arbitrary input x and assume for simplicity of notation that on input x, M uses exactly s qubits and always applies exactly t quantum gates, exactly m of which are measurements, regardless of the observed sequence of measurement outcomes. This can be achieved by padding the computation with gates that do not affect the accept/reject decision of M.

3.3.1 Computation Tree

PP can be characterized as the class of languages consisting of inputs for which the difference of two #P functions exceeds a certain polynomial-time computable threshold. Thus, we would like to express the acceptance probability of M on input x as the ratio of the difference of two #P functions and some polynomial-time computable function. To facilitate the argument, we model the computation of M on input x as a tree, analogous to the usual computation trees one associates with randomized or nondeterministic computations. We can express the final amplitude of a basis state as a linear combination of #P functions, where each #P function counts the number of root-to-leaf paths in the tree that lead to that basis state and have a particular path amplitude. The coefficients in this linear combination are the path amplitudes, which are the products of the transition amplitudes along the path. In order to rewrite the linear combination as a ratio of the above type, we guarantee certain properties of the transition amplitudes in the tree.

- First, our choice of universal set allows us to cancel a common denominator out of any given gate in such a way that the numerators become Gaussian integers. We can make the product of the common denominators the same for all full computation paths, and we will eventually absorb it in the polynomial-time computable threshold function.

- Second, the Gaussian integers, such as the $3 + 4i$ numerator in the F gate, are handled by separating out real and imaginary parts, as well as positive and negative parts, and by multiplicating nodes such that we effectively only need to consider numerators in $\{1, -1, i, -i\}$. By multiplicating nodes we mean that we allow one node to have multiple children representing the same computational basis state. For example, the $3 + 4i$ numerator results in seven children.

We formally define the computation tree for our fixed input x as follows. It has depth t. Each level $\tau = 0, \ldots, t$ represents the state of the quantum tape after the τth gate is applied and before the $(\tau + 1)$st gate is applied. Each node v has five labels:

- $\tau(v) \in \{0, \ldots, t\}$, representing the level of v
- $\mu(v) \in \{0, 1\}^{\leq m}$, representing the sequence of measurement outcomes that leads to v
- $\sigma(v) \in \{0, 1\}^s$, representing a computational basis state
- $\alpha(v) \in \{1, -1, i, -i\}$, representing the numerator of the amplitude of v
- $\beta(v) \in \mathbb{R}^+$, representing the denominator of the amplitude of v

Note that $\alpha(v)$ will be the product of the numerators of the transition amplitudes along the path that leads to v, and similarly for $\beta(v)$. Labels of nodes across a given level need not be unique; if v and u are at the same level and $\sigma(v) = \sigma(u)$ and $\mu(v) = \mu(u)$, then v and u represent interference.

We now define the tree inductively as follows. The root node v is at level $\tau(v) = 0$ and has $\mu(v) = \epsilon$ representing that no measurements have been performed yet, $\sigma(v) = 0^s$ representing the initial state, and $\alpha(v) = \beta(v) = 1$ representing that $|0^s\rangle$ has amplitude 1 initially. Now consider an arbitrary node v. If $\tau(v) = t$ then v is a leaf. Otherwise, v has children at level $\tau(v) + 1$ that depend on the type and operands of $(\tau(v) + 1)$st gate applied given that $\mu(v)$ is observed. Let G denote this gate.

- If $G = H$ then v has two children v_0 and v_1. Suppose G is applied to the jth qubit and let $\sigma(v_0)$ and $\sigma(v_1)$ be obtained from $\sigma(v)$ by setting the jth bit to 0 for $\sigma(v_0)$ and to 1 for $\sigma(v_1)$. Let $\sigma(v)_j$ denote the jth bit of $\sigma(v)$. If $\sigma(v)_j = 0$ then put $\alpha(v_0) = \alpha(v_1) = \alpha(v)$, and if $\sigma(v)_j = 1$ then put $\alpha(v_0) = \alpha(v)$ and $\alpha(v_1) = -\alpha(v)$. In each case put $\beta(v_0) = \beta(v_1) = \sqrt{2}\beta(v)$. The 1 and -1 multipliers for the α-labels correspond to the transition amplitudes of G, except that the common $1/\sqrt{2}$ has been factored out and absorbed in the β-label.

- If $G = F$ then we consider two cases. Suppose G is applied to the jth qubit. If $\sigma(v)_j = 1$ then v has seven children, all with the same σ-label $\sigma(v)$. Three of the children have α-label $\alpha(v)$ and the other four have α-label $i \cdot \alpha(v)$, and all children have β-label $5\beta(v)$. This corresponds to an amplitude of $\frac{3+4i}{5}$, where the numerator $3 + 4i$ has been spread out across multiple children so as to maintain the property that all α-labels are in $\{1, -1, i, -i\}$. If $\sigma(v)_j = 0$ then v has five children, again all with the same σ-label $\sigma(v)$, and now all with the same α-label $\alpha(v)$ and all with β-label $5\beta(v)$. This corresponds to an amplitude of 5/5, so that a common denominator of 5 can be used for all nodes resulting from the application of G.

- If $G = F^\dagger$ then the children of v are constructed as in the case $G = F$ except that the children with α-label $i \cdot \alpha(v)$ now have α-label $-i \cdot \alpha(v)$.

- If $G = I$, CNOT, a query gate, or a measurement gate, then applying G to $|\sigma(v)\rangle$ yields another computational basis state $|\sigma\rangle$, so v has a single child u with $\sigma(u) = \sigma$, $\alpha(u) = \alpha(v)$, and $\beta(u) = \beta(v)$.

For the cases where G is unitary, we put $\mu(u) = \mu(v)$ for all children u. If G is a measurement gate then we put $\mu(u) = \mu(v)\sigma(v)_j$, where u is the unique child of v and j is the index of the qubit measured by G.

Note that the denominator $\beta(v)$ can be written as $5^{f(v)}\sqrt{2}^{h(v)}$, where $f(v)$ denotes the number of F and $F^\dagger$ gates along the path from the root to v, and $h(v)$ the number of H gates. In fact, $f(v)$ and $h(v)$ can be viewed as functions of $\tau(v)$ and $\mu(v)$ only, and we will write $f(\tau, \mu)$ and $h(\tau, \mu)$ accordingly. This is because the sequence of gates that leads to a node v only depends on $\mu(v)$ (and on the fixed input x). The latter reflects the obliviousness of the model to the quantum computation path.

In order to describe how the computation tree reflects the evolution of the quantum tape, we introduce the following notation:

- $V_{\tau,\mu,\sigma,\alpha} = \{v : \tau(v) = \tau,\ \mu(v) = \mu,\ \sigma(v) = \sigma,\ \alpha(v) = \alpha\}$
- $V_{\tau,\mu,\sigma} = \bigcup_\alpha V_{\tau,\mu,\sigma,\alpha}$
- $V_{\tau,\mu} = \bigcup_\sigma V_{\tau,\mu,\sigma}$

Suppose we run M but do not renormalize state vectors after measurements. Then after τ gates have been applied, we have a vector for each sequence of measurement outcomes μ that could have occurred during the first τ steps. The nodes in $V_{\tau,\mu}$ together with their amplitudes

$$\frac{\alpha(v)}{5^{f(\tau,\mu)}\sqrt{2}^{h(\tau,\mu)}}$$

give the vector for μ, since these are exactly the nodes whose computation paths are consistent with the measurement outcomes $\mu_1 \cdots \mu_{|\mu|}$. More precisely, an inductive argument shows that the vector for μ equals

$$\frac{\sum_{v \in V_{\tau,\mu}} \alpha(v)|\sigma(v)\rangle}{5^{f(\tau,\mu)}\sqrt{2}^{h(\tau,\mu)}}.$$

The squared 2-norm of each such vector equals the probability p_μ of observing μ. In particular, at the end of the computation we obtain the following key property.

Claim 1. *For all $\mu \in \{0,1\}^m$,*

$$p_\mu = \frac{\sum_{\sigma \in \{0,1\}^s} \left| \sum_{v \in V_{t,\mu,\sigma}} \alpha(v) \right|^2}{25^{f(t,\mu)} 2^{h(t,\mu)}}.$$

We present a formal proof of Claim 1 in Appendix A.

3.3.2 Machine Construction

With Claim 1 in hand, we now show how to construct a probabilistic machine N running in time $O(t)$ and space $O(s + \log t)$ such that for all inputs x,

- if $\Pr(M \text{ accepts } x) > 1/2$ then $\Pr(N \text{ accepts } x) > 1/2$, and
- if $\Pr(M \text{ accepts } x) < 1/2$ then $\Pr(N \text{ accepts } x) < 1/2$.

This suffices to prove Lemma 2.

We first construct nondeterministic machines M_1, M_{-1}, M_i, M_{-i}, each taking as input a triple (x, μ, σ) where $x \in \{0,1\}^n$, $\mu \in \{0,1\}^m$, and $\sigma \in \{0,1\}^s$. (Recall that most of our notation, such as m and s, is with reference to the particular input x.) For each $\alpha \in \{1, -1, i, -i\}$, M_α will run in time $O(t)$ and space $O(s + \log t)$ and satisfy $\#M_\alpha(x, \mu, \sigma) = |V_{t,\mu,\sigma,\alpha}|$, where $\#M_\alpha(x, \mu, \sigma)$ denotes the number of accepting computation paths of M_α on input (x, μ, σ). Since t is constructible, we can assume without loss of generality that all machines are constructed so as to have exactly 2^g computation paths for some constructible function $g = O(t)$. This allows us to compare numbers of accepting paths to numbers of rejecting paths.

We simply have $M_\alpha(x, \mu, \sigma)$ nondeterministically guess a root-to-leaf path in the computation tree. The only information about the current node v it needs to keep track of is $\sigma(v)$ and $\alpha(v)$, taking space $O(s)$. It keeps a pointer into μ, taking space $O(\log t)$. It determines the correct sequence of gates by simulating the classical part of M, taking $O(t)$ time and $O(s)$ space. When processing a measurement gate G, M_α checks that applying G to the current $\sigma(v)$ yields the next bit of μ. It rejects if not and otherwise continues, using that bit of μ as the measurement outcome. When it reaches a leaf v, M_α checks that $\sigma(v) = \sigma$ and $\alpha(v) = \alpha$ and accepts if so and rejects otherwise. As constructed, M_α has the desired behavior.

Fix $\mu \in \{0,1\}^m$. By Claim 1, the probability of observing μ satisfies

$$\begin{aligned} p_\mu &= \sum_{\sigma\in\{0,1\}^s} \frac{\left|\sum_{v\in V_{t,\mu,\sigma}} \alpha(v)\right|^2}{25^{f(t,\mu)}2^{h(t,\mu)}} \\ &= \sum_{\sigma\in\{0,1\}^s} \frac{\left|\sum_\alpha \alpha\cdot \#M_\alpha(x,\mu,\sigma)\right|^2}{25^{f(t,\mu)}2^{h(t,\mu)}} \\ &= \sum_{\sigma\in\{0,1\}^s} \frac{\left(\sum_\alpha \#M_\alpha(x,\mu,\sigma)^2\right) - \left(\sum_\alpha \#M_\alpha(x,\mu,\sigma)\cdot \#M_{-\alpha}(x,\mu,\sigma)\right)}{25^{f(t,\mu)}2^{h(t,\mu)}} \\ &= \sum_{\sigma\in\{0,1\}^s} \frac{\#M_+(x,\mu,\sigma) - \#M_-(x,\mu,\sigma)}{25^{f(t,\mu)}2^{h(t,\mu)}}, \end{aligned}$$

where M_+ is a nondeterministic machine that guesses $\alpha \in \{1,-1,i,-i\}$ and then runs two copies of M_α, accepting iff both accept, and M_- is a nondeterministic machine that guesses $\alpha \in \{1,-1,i,-i\}$ and then runs a copy of M_α and a copy of $M_{-\alpha}$, accepting iff both accept. We run the copies *in parallel*, keeping them in synch so that they access each bit of μ at the same time. Note that since M_+ and M_- can reject after seeing a single disagreement with μ, the two copies being run will apply the same sequence of gates and thus access each bit of μ at the same time. It follows that both M_+ and M_- need to reference each bit of μ only once. As we show shortly, this is critical for preserving the space bound. Both M_+ and M_- run in time $O(t)$ and space $O(s+\log t)$.

In order to capture the probability of acceptance of M, we would like to sum over all complete sequences of measurement outcomes μ that cause M to accept. We assume without loss of generality that $f(t,\mu)$ and $h(t,\mu)$ are independent of μ, say $f(t,\mu)=f$ and $h(t,\mu)=h$ for all μ, so that the scaling factor $1/25^{f(t,\mu)}2^{h(\mu)}$ can be factored out of this sum. To achieve this, we can modify M so that it counts the number of F and $F^\dagger$ gates and the number of H gates it applies during the computation and then applies some dummy gates at the end to bring the counts up to the fixed values f and h.

We construct nondeterministic machines N_+ and N_-, both running in time $O(t)$ and space $O(s+\log t)$, such that

$$\#N_+(x) = \sum_{\substack{\mu\,\in\,\{0,1\}^m \text{ causing} \\ M \text{ to accept}}} \; \sum_{\sigma\in\{0,1\}^s} \#M_+(x,\mu,\sigma)$$

and

$$\#N_-(x) = \sum_{\substack{\mu\,\in\,\{0,1\}^m \text{ causing} \\ M \text{ to accept}}} \; \sum_{\sigma\in\{0,1\}^s} \#M_-(x,\mu,\sigma).$$

We have $N_+(x)$ run $M_+(x,\mu,\sigma)$ for a nondeterministically guessed $\mu \in \{0,1\}^m$ and $\sigma \in \{0,1\}^s$ and accept iff M_+ accepts and μ causes M to accept, and similarly for N_-. Since every accepting execution of M_+ or M_- follows an execution of M with measurement outcomes μ, we know at the end whether μ causes M to accept.

However, letting N_+ just nondeterministically guess μ and σ and then run $M_+(x,\mu,\sigma)$ does not work because it takes too much space to store μ. Since M_+ and M_- were constructed in such a way that each bit of μ is only referenced once, we can nondeterministically guess each μ_j when needed and overwrite the previous μ_{j-1}. The space usage of σ is not an issue, so σ can be guessed and stored at any time. Constructed in this way, N_+ and N_- have the desired properties.

It follows that the probability M accepts x is

$$\frac{\#N_+(x) - \#N_-(x)}{25^f 2^h}.$$

Thus,

- if $\Pr(M \text{ accepts } x) > 1/2$ then $\#N_+(x) - \#N_-(x) > 25^f 2^{h-1}$, and
- if $\Pr(M \text{ accepts } x) < 1/2$ then $\#N_+(x) - \#N_-(x) < 25^f 2^{h-1}$.

We can now use a standard technique to obtain the final PTISP simulation N. We assume without loss of generality that $h \geq 1$. Recall that we assume N_+ and N_- each always have exactly 2^g computation paths for some constructible function $g = O(t)$. By nondeterministically picking N_+ or N_- to run, and flipping the answer if N_- was chosen, we get $\big(\#N_+(x) - \#N_-(x)\big) + 2^g$ accepting computation paths. We can generate 2^{g+1} dummy computation paths, exactly $2^g + 25^f 2^{h-1}$ of which reject, to shift the critical number of accepting paths to exactly half the total number of computation paths. To do this very time- and space-efficiently, we take advantage of our use of a universal set, which gives the number of rejecting dummy paths a simple form. We have N nondeterministically guess $g+1$ bits; if the first bit is 0 it rejects, and otherwise it ignores the next $h-1$ bits, groups the next $6f$ bits into groups of 6 forming a number in $\{0, \ldots, 31\}$, accepts if any group is at least 25, and otherwise rejects iff the remaining guess bits are 0. Since t is constructible, we can take $f = O(t)$ and $h = O(t)$ to be constructible functions so that N can compute f and h without affecting the complexity parameters.

As constructed, N runs in time $O(t)$ and space $O(s + \log t)$ and accepts x with probability greater than 1/2 if $x \in L$ and with probability less than 1/2 if $x \notin L$. This finishes the proof of Lemma 2.

3.4 Remarks

Notice that the proof of Lemma 2 shows that the unbounded-error version of $\mathrm{BQTISP}''(t, s)$ is contained in $\mathrm{PTISP}(t,\ s + \log t)$. Since Theorem 1 only operates at the granularity of polynomial time and space bounds, it would suffice to have Lemma 5 show that $\mathrm{BQTISP}'(t, s)$ is contained in the unbounded-error version of $\mathrm{BQTISP}''(t^{1+o(1)},\ s + t^{o(1)})$. This allows us to prove Theorem 1 under a more relaxed definition of $\mathrm{BQTISP}(t, s)$:

- We could relax the error probability to $1/2 - 1/\mathrm{poly}(t)$ and relax the time for computing the first k bits of the amplitudes to $2^{o(k)}$, since the overhead in computing amplitudes would still be subpolynomial and the Solovay-Kitaev algorithm could still produce $1/\mathrm{poly}(t)$-approximations, for arbitrarily high degree polynomials, in $t^{o(1)}$ time. The argument in Lemma 5 still proves that the error probability remains less than 1/2 in this case.
- Alternatively, we could keep the amplitude efficiency at $\mathrm{poly}(k)$ and relax the error probability to $1/2 - 1/2^{t^{o(1)}}$; then the Solovay-Kitaev algorithm would need to compute $1/2^{t^{o(1)}}$-approximations, which would still only take $t^{o(1)}$ time.

A natural goal is to strengthen Lemma 2 to unbounded-error quantum algorithms; the problem is that the error probability gets degraded during the approximation process of Lemma 5 and thus needs to be bounded away from $1/2$ by a nonnegligible amount to begin with.

Finally, we remark that nothing prevents our proof of Lemma 2 from carrying over to any reasonable model of quantum computation that is nonoblivious to the quantum computation path. In this case, the sequence of gates leading to a node v in the computation tree does not only depend on $\mu(v)$, but this fact does not present a problem for our proof. However, the proof becomes more technical since, e.g., approximating each local unitary operation may lead to overall nonunitary evolution since the local operations are themselves applied in a superposition. These complications arise for the same reason as the unnatural conditions on the transition function in the models from [3] and [18]. We feel that working out the details of such a result would not be well-motivated since the currently envisioned realizations of quantum computers do not support such nonoblivious behavior.

4 Conclusion

Several questions remain open regarding time-space lower bounds on quantum models of computation. An obvious goal is to obtain a quantitative improvement to our lower bound. It would be nice to get a particular constant $c > 1$ such that MajMajSAT cannot be solved by quantum algorithms running in n^c time and subpolynomial space. The lower bound of Allender et al. does yield this; however, the constant c is very close to 1, and determining it would require a complicated analysis involving constant-depth threshold circuitry for iterated multiplication [10]. Perhaps there is a way to remove the need for this circuitry in the quantum setting.

A major goal is to prove quantum time-space lower bounds for problems that are simpler than MajMajSAT. Ideally we would like lower bounds for satisfiability itself, although lower bounds for its cousins in PH and $\oplus$P would also be very interesting. The difficulty in obtaining such lower bounds arises from the fact that we know of no simulations of quantum computations in these classes. The known time-space lower bounds for satisfiability and related problems follow the indirect diagonalization paradigm, which involves assuming the lower bound does not hold and then deriving a contradiction with a direct diagonalization result. For example, applying this paradigm to quantum algorithms solving Σ_ℓSAT would entail assuming that Σ_ℓSAT has an efficient quantum algorithm. Since Σ_ℓSAT is complete for the class Σ_ℓP under very efficient reductions, this hypothesis gives a general simulation of the latter class on quantum algorithms. To reach a contradiction with a direct diagonalization result, we seem to need a way to convert these quantum computations back into polynomial-time hierarchy computations.

Strengthening Corollary 1 to MajSAT instead of MajMajSAT may currently be within reach. Recall that the result of [2] only needs the following two types of hypotheses to derive a contradiction:

- MajMajSAT $\in$ PTIME(n^c), and
- MajSAT $\in$ PTISP$(n^d, n^{1-\epsilon})$.

Under the hypothesis MajSAT $\in$ BQTISP$(n^{1+o(1)}, n^{1-\epsilon})$, Lemma 2 yields the second inclusion but not the first. One can use the hypothesis to replace the second majority quantifier of a MajMajSAT formula with a quantum computation. However, we do not know how to use the hypothesis again to remove the first majority quantifier, because the hypothesis only applies to majority-quantified *deterministic* computations. Fortnow and Rogers [8] prove that $\mathrm{PP}^{\mathrm{BQP}} = \mathrm{PP}$, and their proof shows how to absorb the "quantumness" into the majority quantifier so that we *can* apply the hypothesis again. However, their proof critically uses time-expensive amplification and is not efficient enough to yield a lower bound for MajSAT via the result of [2]. It might be possible to exploit the space bound to obtain a more efficient inclusion. It might also be possible to exploit more special properties of the construction in [2] to circumvent the need for the amplification component.

Acknowledgments

We would like to thank Scott Diehl for helpful comments.

References

[1] L. Adleman, J. DeMarrais, and M. Huang. Quantum Computability. *SIAM Journal on Computing*, 26(5): 1524-1540, 1997.

[2] E. Allender, M. Koucky, D. Ronneburger, S. Roy, and V. Vinay. Time-Space Tradeoffs in the Counting Hierarchy. In *Proceedings of the 16th IEEE Conference on Computational Complexity*, pages 295-302, 2001.

[3] E. Bernstein and U. Vazirani. Quantum Complexity Theory. *SIAM Journal on Computing*, 26(5): 1411-1473, 1997.

[4] C. M. Dawson and M. A. Nielsen. The Solovay-Kitaev Algorithm. arXiv:quant-ph/0505030

[5] S. Diehl and D. van Melkebeek. Time-Space Lower Bounds for the Polynomial-Time Hierarchy on Randomized Machines. *SIAM Journal on Computing*, 36: 563-594, 2006.

[6] L. Fortnow. Time-Space Tradeoffs for Satisfiability. *Journal of Computer and System Sciences*, 60(2): 337-353, 2000.

[7] L. Fortnow, R. Lipton, D. van Melkebeek, and A. Viglas. Time-Space Lower Bounds for Satisfiability. *Journal of the ACM*, 52: 835-865, 2005.

[8] L. Fortnow and J. Rogers. Complexity Limitations on Quantum Computation. *Journal of Computer and System Sciences*, 59(2): 240-252, 1999.

[9] L. K. Grover. A Fast Quantum Mechanical Algorithm for Database Search. In *Proceedings of the 28th ACM Symposium on Theory of Computing*, pages 212-219, 1996.

[10] W. Hesse. Division is in Uniform TC^0. In *Proceedings of the 28th International Colloquium On Automata, Languages, and Programming*, pages 104-114, 2001

[11] A. Kitaev. Quantum Computations: Algorithms and Error Correction. *Russian Mathematical Surveys*, 52(6): 1191-1249, 1997.

[12] A. Kitaev, A. Shen, and M. Vyalyi. *Classical and Quantum Computation.* American Mathematical Society, 2002.

[13] D. van Melkebeek. A Survey of Lower Bounds for Satisfiability and Related Problems. *Foundations and Trends in Theoretical Computer Science*, 2: 197-303, 2007.

[14] M. A. Nielsen and I. L. Chuang. *Quantum Computation and Quantum Information.* Cambridge University Press, 2000.

[15] P. Shor. Polynomial-Time Algorithms for Prime Factorization and Discrete Logarithms on a Quantum Computer. *SIAM Journal on Computing*, 26(5): 1484-1509, 1997.

[16] S. Toda. PP is as Hard as the Polynomial Time Hierarchy. *SIAM Journal on Computing*, 20(5): 865-877, 1991.

[17] J. Watrous. On the Complexity of Simulating Space-Bounded Quantum Computations. *Computational Complexity*, 12: 48-84, 2003.

[18] J. Watrous. Space-Bounded Quantum Complexity. *Journal of Computer and System Sciences*, 59(2): 281-326, 1999.

[19] R. Williams. Inductive Time-Space Lower Bounds for SAT and Related Problems. *Computational Complexity*, 15(4): 433-470, 2006.

[20] R. Williams. Time-Space Tradeoffs for Counting NP Solutions Modulo Integers. In *Proceedings of the 22nd IEEE Conference on Computational Complexity*, pages 70-82, 2007.

A Postponing Measurements

In this appendix we describe a framework for analyzing quantum algorithms with intermediate measurements by implicitly postponing the measurements and tracking the unitary evolution of the resulting purification. We stress that we are doing so for reasons of analysis only; our actual simulations do not involve postponing measurements. This framework facilitates the proofs of Claim 1 and Lemma 5. We first describe the common framework and then use it for those two proofs.

Consider a quantum Turing machine M running in time t and space s. We fix an arbitrary input x and assume for simplicity of notation that on input x, M uses exactly s qubits and always applies exactly t quantum gates, exactly m of which are measurements, regardless of the observed sequence of measurement outcomes. This can be achieved by padding the computation with gates that do not affect the accept/reject decision of M.

We conceptually postpone the measurements in the computation by

- introducing m ancilla qubits initialized to all 0's,
- replacing the ith measurement on each classical computation path by an operation that entangles the ith ancilla qubit with the qubit being measured (by applying a CNOT to the ancilla with the measured qubit as the control), and
- measuring the m ancilla qubits at the end.

In the τth step of the simulation, we apply a unitary operation U_τ on a system of $s+m$ qubits, where U_τ acts independently on each of the subspaces corresponding to distinct sequences of measurement outcomes that can be observed before time step τ. More precisely, consider the set of $\mu \in \{0,1\}^{\leq m}$ such that given that μ is observed, the τth gate is applied after μ is observed but not after the $(|\mu|+1)$st measurement gate is applied. Let $\mathcal{U}_\tau$ be the set of μ such that the τth gate is unitary, and let $\mathcal{M}_\tau$ be the set of μ such that the τth gate is a measurement. For $\nu \in \{0,1\}^m$, let P_ν denote the projection on the state space of the ancilla qubits to the one-dimensional subspace spanned by $|\nu\rangle$.

For $\mu \in \mathcal{U}_\tau$, let $G_{\tau,\mu}$ denote the unitary operator on the state space of s qubits induced by the τth gate applied given that μ is observed. Then U_τ acts as $G_{\tau,\mu} \otimes I$ on the range of $I \otimes P_{\mu 0^{m-|\mu|}}$. For each $\mu \in \mathcal{M}_\tau$, U_τ applies an entangling operation $E_{\tau,\mu}$ that acts only on the range of $I \otimes (P_{\mu 0^{m-|\mu|}} + P_{\mu 1 0^{m-1-|\mu|}})$. The behavior of U_τ on the remaining subspaces does not matter; we can set it arbitrarily to the identity operator. Thus,

$$U_\tau = \Big(\sum_{\mu \in \mathcal{U}_\tau} G_{\tau,\mu} \otimes P_{\mu 0^{m-|\mu|}} \Big) + \Big(\sum_{\mu \in \mathcal{M}_\tau} E_{\tau,\mu} \Big) + R,$$

where R is a term that expresses the behavior on the remaining subspaces.

It is well-known, and can be verified from first principles, that the probability of observing any sequence of measurement outcomes $\mu \in \{0,1\}^m$ when M is run equals the probability of observing μ after the evolution $U = U_t U_{t-1} \cdots U_2 U_1$ with all of the ancilla qubits initialized to 0. That is, $\Pr(\mu \text{ observed}) = \big|\big|(I \otimes P_\mu) U |0^{s+m}\rangle\big|\big|^2$.

We next prove Claim 1 and Lemma 5. These proofs both use the above framework but are otherwise independent of each other.

A.1 Proof of Claim 1

Recall that we have a tree expressing the computation of M on the fixed input x, and we wish to show that the probability of observing any complete sequence of measurement outcomes $\mu \in \{0,1\}^m$ satisfies

$$p_\mu = \frac{\sum_{\sigma \in \{0,1\}^s} \big| \sum_{v \in V_{t,\mu,\sigma}} \alpha(v) \big|^2}{25^{f(t,\mu)} 2^{h(t,\mu)}}.$$

Consider the above postponed measurement framework. The state of the system after τ steps is given by $U_\tau \cdots U_1 |0^{s+m}\rangle$. We can also write this state as a sum of contributions from all nodes in the τth level of the tree. More precisely, we claim that

$$U_\tau \cdots U_1 |0^{s+m}\rangle = \sum_{v \in V_\tau} |\psi(v)\rangle, \tag{1}$$

where $V_\tau = \bigcup_\mu V_{\tau,\mu}$ and for each node v,

$$|\psi(v)\rangle = \frac{\alpha(v)}{\beta(v)} |\sigma(v)\mu(v)0^{m-|\mu(v)|}\rangle = \frac{\alpha(v)|\sigma(v)\mu(v)0^{m-|\mu(v)|}\rangle}{5^{f(\tau(v),\mu(v))}\sqrt{2}^{h(\tau(v),\mu(v))}}.$$

Note that $|\psi(v)\rangle$ is the basis state of v multiplied by its amplitude, with the ancilla qubits set to indicate the sequence of measurement outcomes that leads to v.

We argue that the decomposition (1) holds by induction on $\tau = 0, \ldots, t$. The base case $\tau = 0$ is trivial. For $\tau > 0$, by induction it suffices to show that for each node $v \in V_{\tau-1}$, $U_\tau|\psi(v)\rangle = \sum_{u \in c(v)} |\psi(u)\rangle$, where $c(v)$ denotes the set of children of v. There are two cases. If $\mu(v) \in \mathcal{U}_\tau$ then it can be verified directly from the construction of the tree that

$$U_\tau|\psi(v)\rangle = \big(G_{\tau,\mu(v)} \otimes P_{\mu(v)0^{m-|\mu(v)|}}\big)|\psi(v)\rangle = \sum_{u \in c(v)} |\psi(u)\rangle.$$

If $\mu(v) \in \mathcal{M}_\tau$ then it can be directly verified that $U_\tau|\psi(v)\rangle = E_{\tau,\mu(v)}|\psi(v)\rangle = |\psi(u)\rangle$, where u is the child of v. This completes the induction step.

Given decomposition (1) for $\tau = t$, we obtain that for all $\mu \in \{0,1\}^m$,

$$p_\mu = \big|\big|(I \otimes P_\mu)U|0^{s+m}\rangle\big|\big|^2 = \Big|\Big|\sum_{v \in V_{t,\mu}} |\psi(v)\rangle\Big|\Big|^2 = \frac{\sum_{\sigma \in \{0,1\}^s} \big|\sum_{v \in V_{t,\mu,\sigma}} \alpha(v)\big|^2}{25^{f(t,\mu)}2^{h(t,\mu)}}.$$

A.2 Proof of Lemma 5

Consider a language $L \in \mathrm{BQTISP}'(t,s)$ and an associated quantum Turing machine M', and fix an arbitrary input x. We assume as above that on input x, M' uses exactly s qubits and always applies exactly t gates, exactly m of which are measurements. We transform M' into a machine M'' running in time $t \cdot \mathrm{polylog}(t)$ and space $s + \mathrm{polylog}(t)$ accepting L with error probability bounded away from 1/2 by a constant. By standard amplification techniques, the error probability can be made at most 1/3, so $L \in \mathrm{BQTISP}''(t \cdot \mathrm{polylog}(t),\ s + \mathrm{polylog}(t))$.

Using Lemma 4, we have M'' run M' but replace each single-qubit unitary gate with a $1/20t$-approximation consisting of at most polylog(t) gates from the set $\{F, F^\dagger, H\}$. The time and space overhead is polylog(t), so M'' runs in time $t \cdot \mathrm{polylog}(t)$ and space $s + \mathrm{polylog}(t)$ and still operates on s qubits. We now show that the probability M'' accepts x differs from the probability M' accepts x by at most 1/10. This suffices to prove the lemma.

Let $U' = U'_t \cdots U'_1$ be the evolution on the state space of $s + m$ qubits obtained by implicitly postponing measurements in the computation of M' as described above, and let the notation $\mathcal{U}_\tau$ and $G_{\tau,\mu}$ be as above for this computation. Since the value of μ uniquely determines whether M' accepts, we have that $\Pr(M' \text{ accepts}) = \big|\big|PU'|0^{s+m}\rangle\big|\big|^2$, where P denotes sum of $I \otimes P_\mu$ over all μ consistent with acceptance.

Now let $U'' = U''_t \cdots U''_1$ be the same evolution as U' but where each unitary operation $G_{\tau,\mu}$ is replaced by the operation $\widetilde{G}_{\tau,\mu}$ that uses the $1/20t$-approximation of the τth gate M' applies given that μ is observed, found by the Solovay-Kitaev algorithm. Since multiplying by global phase factors does not affect a computation, we can assume that the approximation used in $\widetilde{G}_{\tau,\mu}$ is at distance at most $1/20t$ from the original gate of M'. Tensoring with the identity does not change the 2-norm of an operator, so we also have $\big|\big|\widetilde{G}_{\tau,\mu} - G_{\tau,\mu}\big|\big| \leq 1/20t$. Now since U'' is equivalent to the postponed measurement transformation applied to M'', we have $\Pr(M'' \text{ accepts}) = \big|\big|PU''|0^{s+m}\rangle\big|\big|^2$.

By standard applications of the triangle inequality (see Box 4.1 in [14]), we have that

$$\big|\Pr(M'' \text{ accepts}) - \Pr(M' \text{ accepts})\big| \ \leq\ 2\big|\big|U'' - U'\big|\big| \ \leq\ 2\sum_{\tau=1}^{t} \big|\big|U''_\tau - U'_\tau\big|\big|.$$

Thus in order to show that the acceptance probability of M' and M'' on input x differ by at most $1/10$, it suffices to show that $\left|\left|U''_\tau - U'_\tau\right|\right| \leq 1/20t$ for all τ. The latter holds since for any unit vector $|\psi\rangle$ in the state space of $s+m$ qubits, we have

$$\begin{aligned}\left|\left|(U''_\tau - U'_\tau)|\psi\rangle\right|\right|^2 &= \left|\left| \sum_{\mu \in \mathcal{U}_\tau} \Big((\widetilde{G}_{\tau,\mu} \otimes P_{\mu 0^{m-|\mu|}}) - (G_{\tau,\mu} \otimes P_{\mu 0^{m-|\mu|}}) \Big) |\psi\rangle \right|\right|^2 \\ &= \sum_{\mu \in \mathcal{U}_\tau} \left|\left| \Big((\widetilde{G}_{\tau,\mu} - G_{\tau,\mu}) \otimes I \Big) \big(I \otimes P_{\mu 0^{m-|\mu|}}\big)|\psi\rangle \right|\right|^2 \\ &\leq \sum_{\mu \in \mathcal{U}_\tau} \Big(\frac{1}{20t}\Big)^2 \left|\left| \big(I \otimes P_{\mu 0^{m-|\mu|}}\big)|\psi\rangle \right|\right|^2 \\ &\leq \Big(\frac{1}{20t}\Big)^2.\end{aligned}$$

B Decomposing Quantum Gates

In this appendix we prove Lemma 3, which follows from results proven in Chapter 4 of [14]. We include a proof for reasons of completeness.

For the nontrivial inclusion, consider a language $L \in \mathrm{BQTISP}(t,s)$ and an associated quantum Turing machine M. We convert M into another machine M' by replacing each application of a library gate U with a sequence of gates that effects the same operation as U, where each either is CNOT or acts on only one qubit. Then M' accepts an input x with the same probability as M, so to show that $L \in \mathrm{BQTISP}'(t,s)$ we just need to check that the efficiency parameters are only affected by constant factors and that the matrix entries of the new gates are still efficiently computable in the required sense. The time parameter clearly only goes up by a constant factor that depends on the gates in the library of M.

The transformation is done in two steps and uses results proven in Sections 4.3 and 4.5 of [14]. First, it is shown in [14] that every unitary gate can be decomposed as the product of unitary gates each of which acts nontrivially on only two computational basis states (two-level gates). Applying this transformation to the library gates associated with the family does not affect s. The matrix entries of these two-level gates are obtained via standard math operations from the matrix entries of the original gates and are thus efficiently computable.

Second, it is shown in [14] that each two-level gate can be decomposed into a product of CNOTs and single-qubit gates. This transformation can be done in three steps.

- First, a two-level gate can be decomposed into a product of operations each of which is a controlled single-qubit operation that conditions on many qubits. This is done by using controlled X gates to interchange adjacent computational basis states in a Gray code order so that the two basis states acted on by the two-level gate differ only in a single qubit. Then a controlled single-qubit gate is used to carry out the nontrivial 2×2 submatrix of the two-level gate, and then the basis states are mapped back to their original values (Figure 4.16 in [14]). This does not increase the number of qubits, and all entries in these gates are 0 or 1 or come from the two-level gate, and are hence efficiently computable.
- Second, each large controlled operation can be reduced to X gates, Toffoli gates, and a controlled single-qubit gate that conditions on one qubit (Figure 4.10 in [14]). To accomplish

this, X gates are first used on some of the control qubits to make the controlled operation condition on all qubits being 1. Then these control qubits are ANDed together into some ancilla qubits using a series of Toffoli gates, and the heart of the operation is carried out by a controlled single-qubit gate that conditions on the ancilla qubit holding the AND of the original control qubits. The ANDing operations are reversed so that the ancilla qubits are reset to 0 and can hence be reused and only increase the number of qubits by an additive constant.

- Third, the Toffoli and controlled single-qubit gates can be implemented with special-purpose circuits using only CNOTs and single-qubit gates (Figures 4.9 and 4.6 in [14]). This does not increase the number of qubits, and the matrix entries in this implementation of a controlled-U gate are obtained from the matrix entries for U via standard math operations and are thus efficiently computable.

This finishes the proof of Lemma 3.